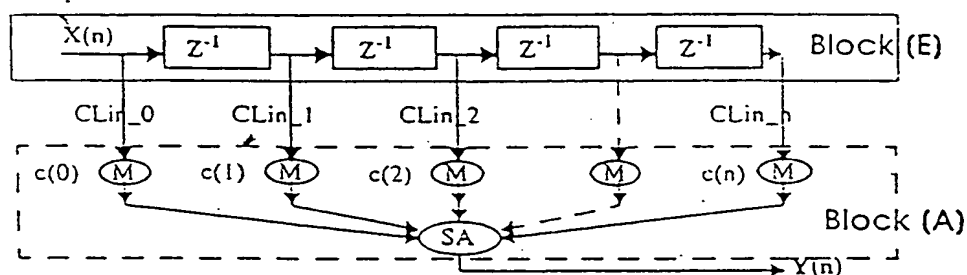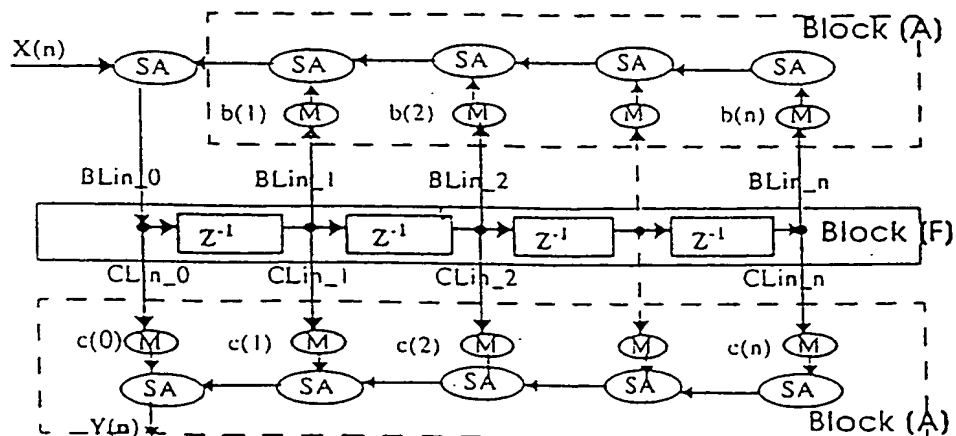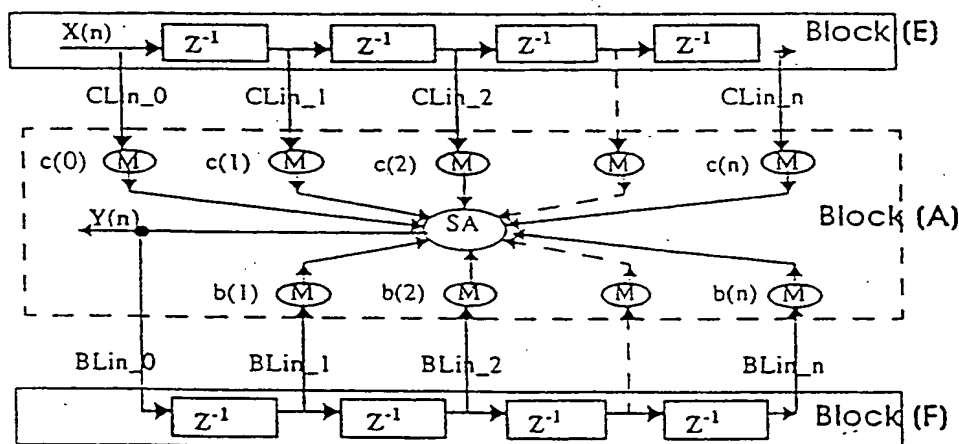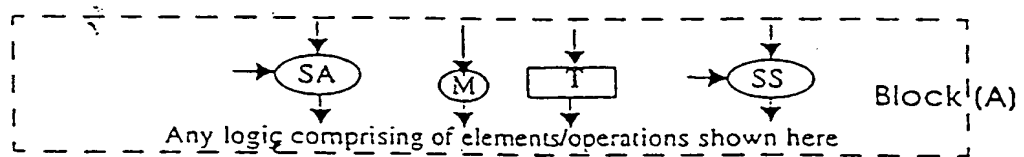FIGURE 1. Field of invention.

FIR filter

IIR filter

Other Application (Combination, sequential logic minimization)

Block (A)

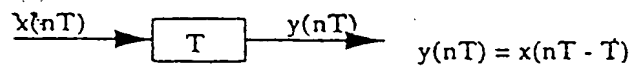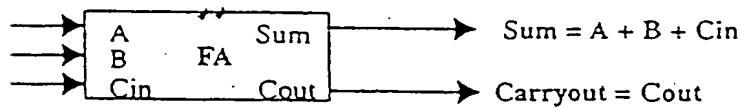Any logic comprising of elements/operations shown here

FIR/IIR filter equation

$$Y(z) = X(z) [c(0) + c(1) Z^{-1} + c(2) Z^{-2} + c(3) Z^{-2} + ... + c(n) Z^{-n}] \qquad ....\text{FIR Eq}$$

$$Y(z) = X(z) \frac{[c(0) + c(1) Z^{-1} + c(2) Z^{-2} + c(3) Z^{-2} + ... + c(n) Z^{-n}]}{[1 - (b(1) Z^{-1} + b(2) Z^{-2} + b(3) Z^{-2} + ... + b(m) Z^{-m})]} \qquad ....\text{IIR Eq}$$

where X(z)-input signal, $Z^{-1} * X(z)$ - delayed signal by one delay, Y(z)-output signal
c(0), c(1), c(2)........c(n), b(1), b(2)......b(m) are integer coefficients values.

FIGURE 2. Bit Serial Elements/components

Unit Delay

$x(nT)$ → [T] → $y(nT)$ $y(nT) = x(nT - T)$

Full adder

[A   Sum
  B FA
  Cin   Cout]   Sum = A + B + Cin

Carryout = Cout

Full subtractor

[A(+)   Diff
  B(-) FS
  Cin   Cout]   Diff = A - B - Cin

Borrow-out = Cout

Serial adder

$x_1(nt)$, $x_2(nT)$ → (SA) → $y(nT)$ $y(nT) = x_1(nT) + x_2(nT)$

Serial subtractor

$x_1(nt)$, $x_2(nT)$ → (SS (-)) → $y(nT)$ $y(nT) = x_1(nT) - x_2(nT)$

Multiplier

m → (M), $x(nT)$ → (M) → $y(nT)$ $y(nT) = m * x(nT)$
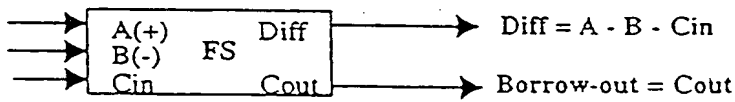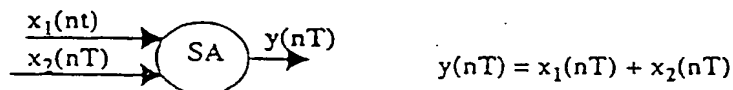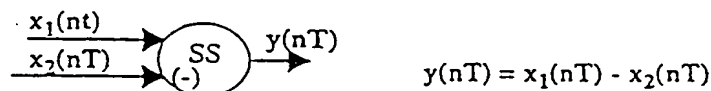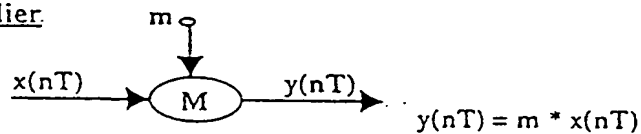
Delay

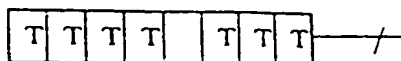input Frame size = X bits (e.g input is 1010101 or X=7 bits)

To store X bit frame, number of T element used is X or 7 in present case

Serial one bit Input
Frame size (X bits) → [Z⁻¹] → Serial one bit Output
Frame size (X bits)

$Eq\ y(n) = x(n-1)$

Serial one bit Input
Frame size (X bits) → OR → [T | T | T | T | T | T | T]

FIGURE 3. Explanation about components used

<u>Unit Delay</u>

$x(nT)$ (0101011)   → D   T   Q   (01010110) → output

clock   → CLK
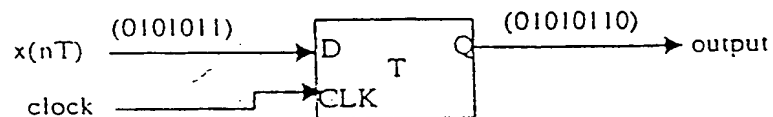
Input frame Input pattern (0101011) is coming serially at $x(nT)$ pin at clock rate specified on clock pin
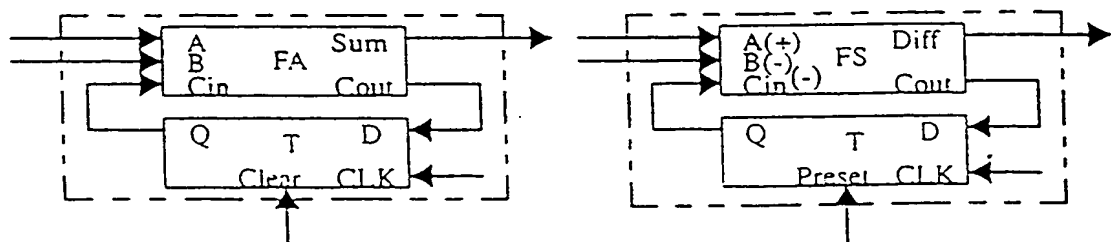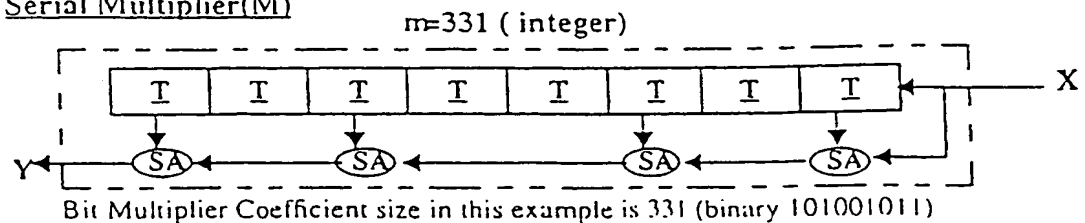
<u>Full adder (FA) / Full subtractor (FS)</u>

binary addition/subtraction components is realized using following truth table

Truth table - Full adder

A	B	Cin	Z	Co
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table - Full Subtractor

A	B	Cin	Z	Co
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

<u>Serial Adder(SA) / Subtractor (SS)</u>

Serial Adder: inputs A, B, Cin into FA producing Sum and Cout; feedback through Q T D with Clear, CLK.

Serial Subtractor: inputs A(+), B(-), Cin(-) into FS producing Diff and Cout; feedback through Q T D with Preset, CLK.

<u>Serial Multiplier(M)</u>

m=331 (integer)

T T T T T T T T ← X

Y ← SA ← SA ← SA ← SA ←

Bit Multiplier Coefficient size in this example is 331 (binary 101001011)

SUBSTITUTE SHEET (RULE 26)

FIGURE 4. Bit Serial Implementation of FIR Filter

Implementation 1
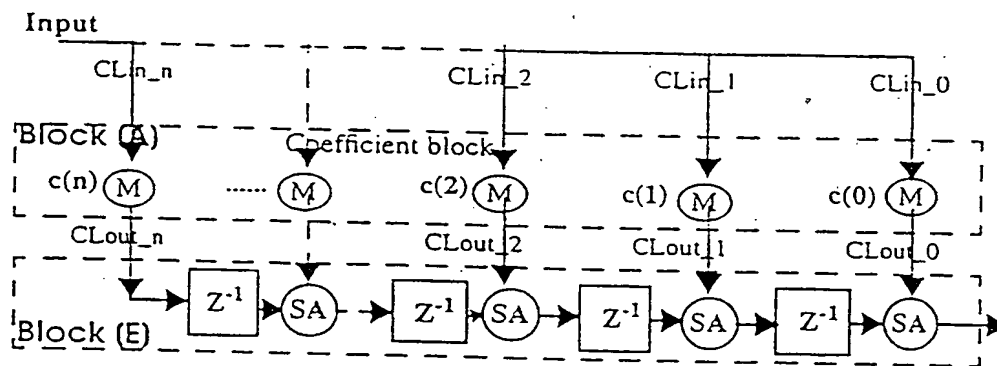
Realization of coefficient using share-able multiplier (coeff. = 3,11)
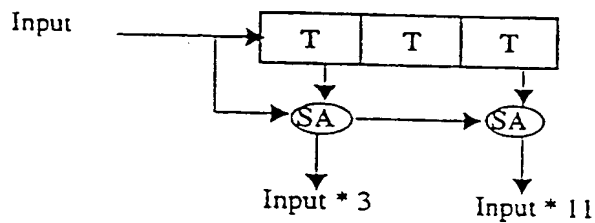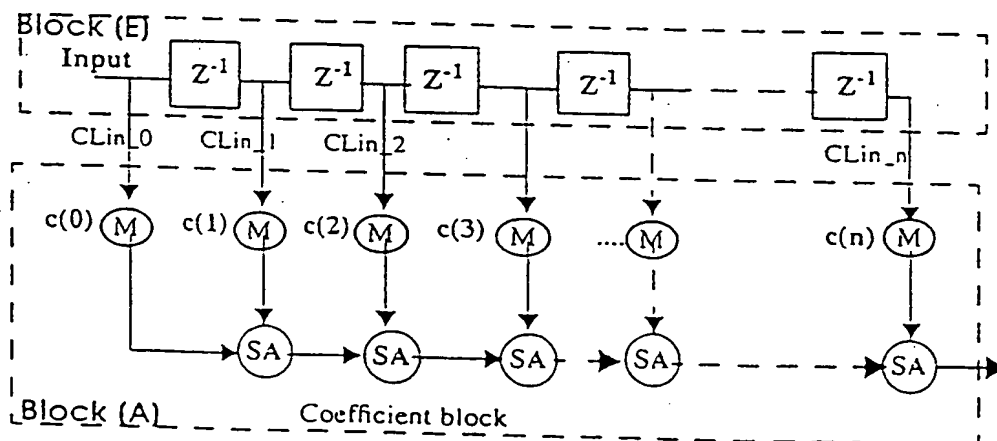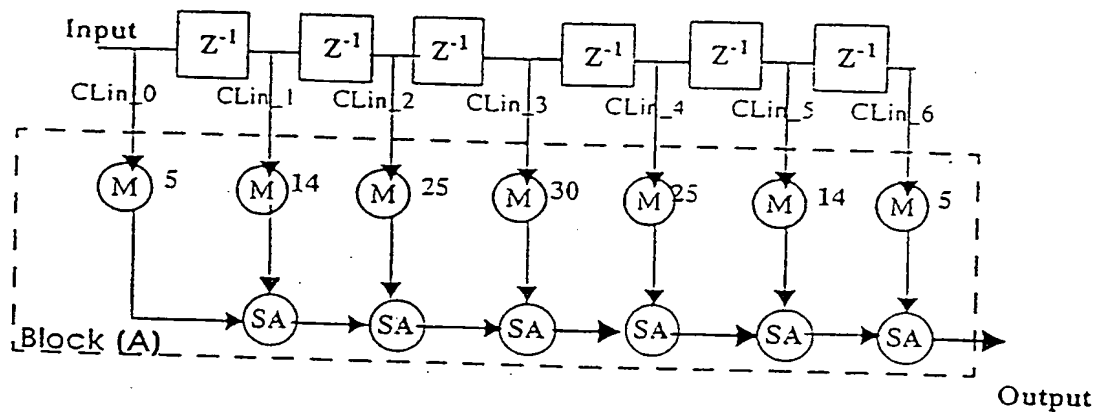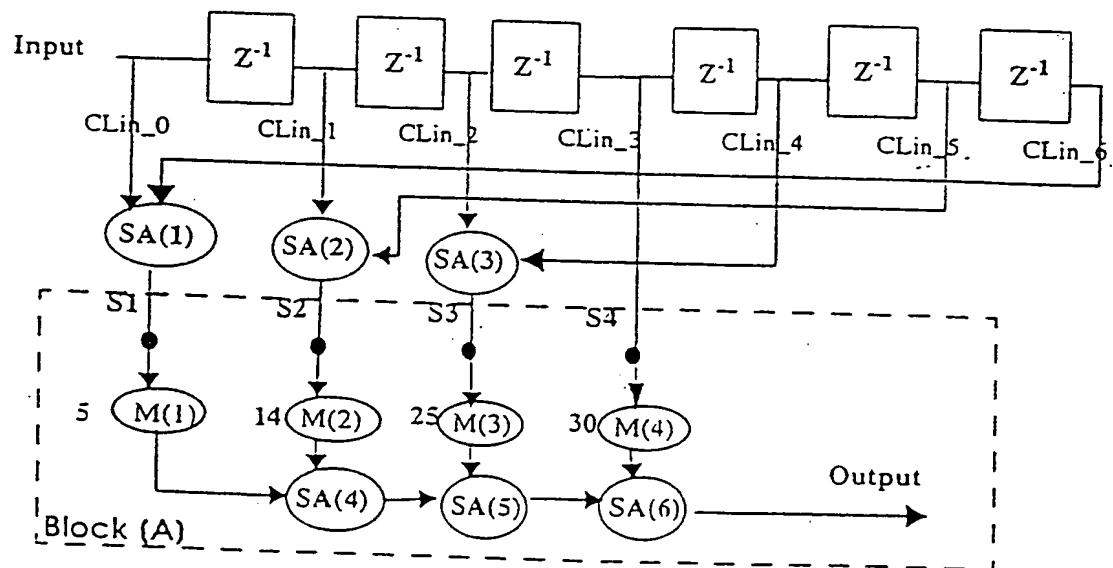

Implementation 2

FIGURE 5. Example FIR Filter

FIGURE 6. An Existing Minimization Technique

Begin: Using the property of symmetrical coefficient

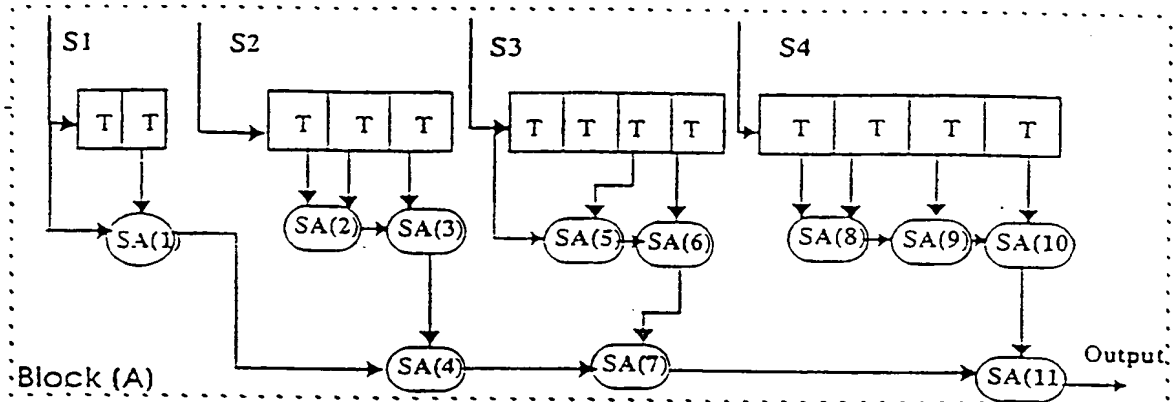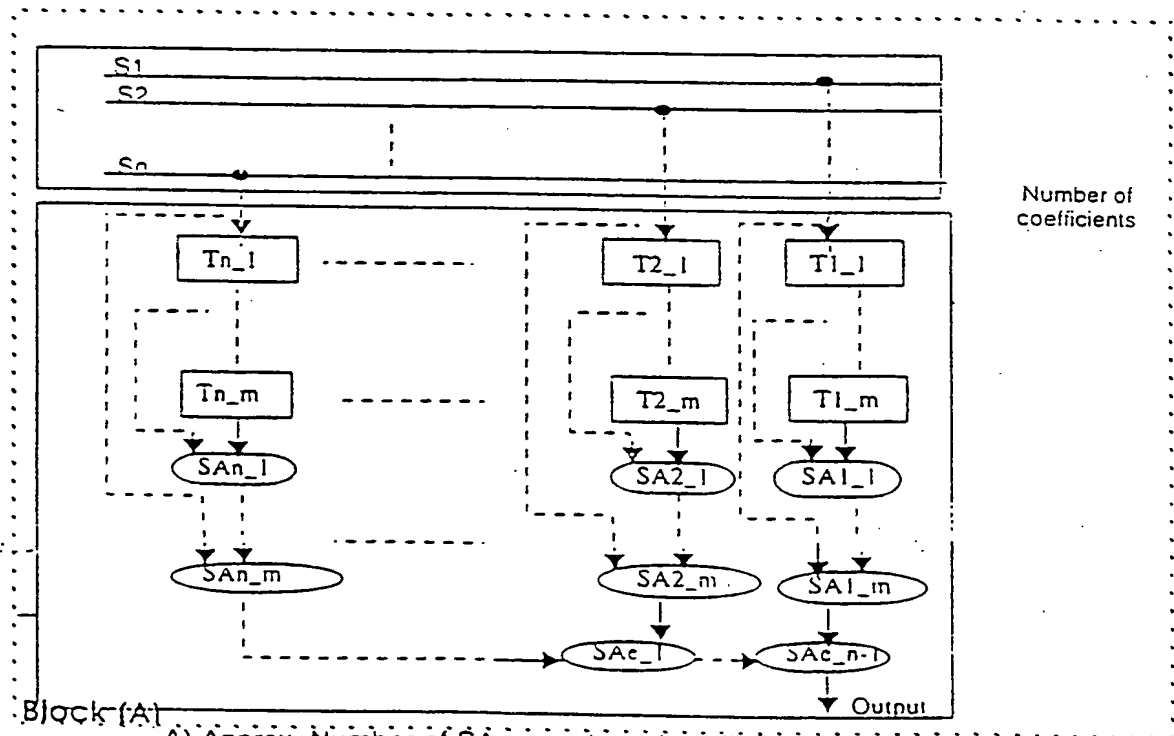FIGURE 7. The "Existing Implementation" of the Coefficient Block [A]

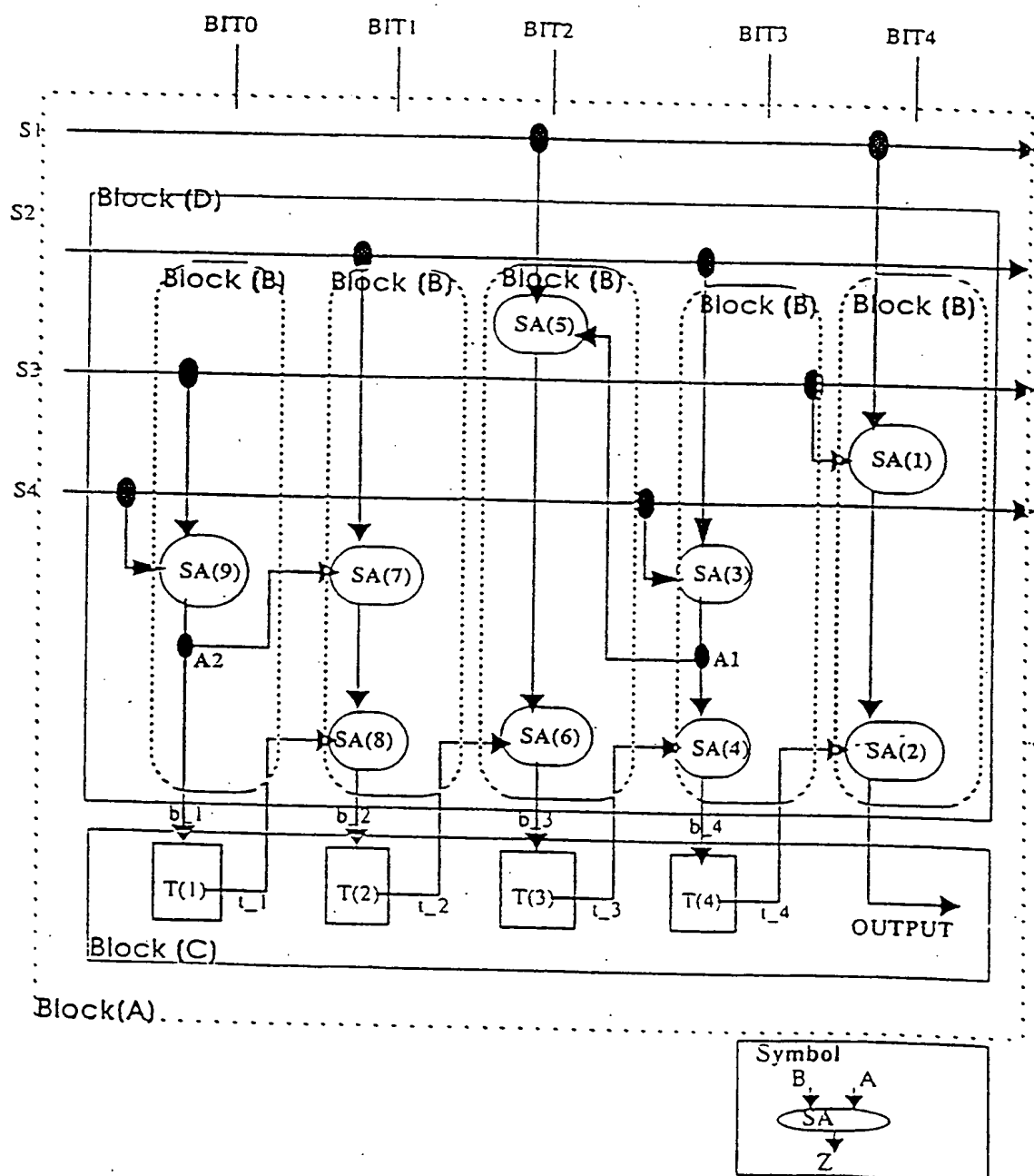FIGURE 8. Generalized structures for "Existing Methods & minimizations"

A) Approx. Number of SA = number of coefficient * max coeff size /2)

B) Flip-flop (T) are not sharable
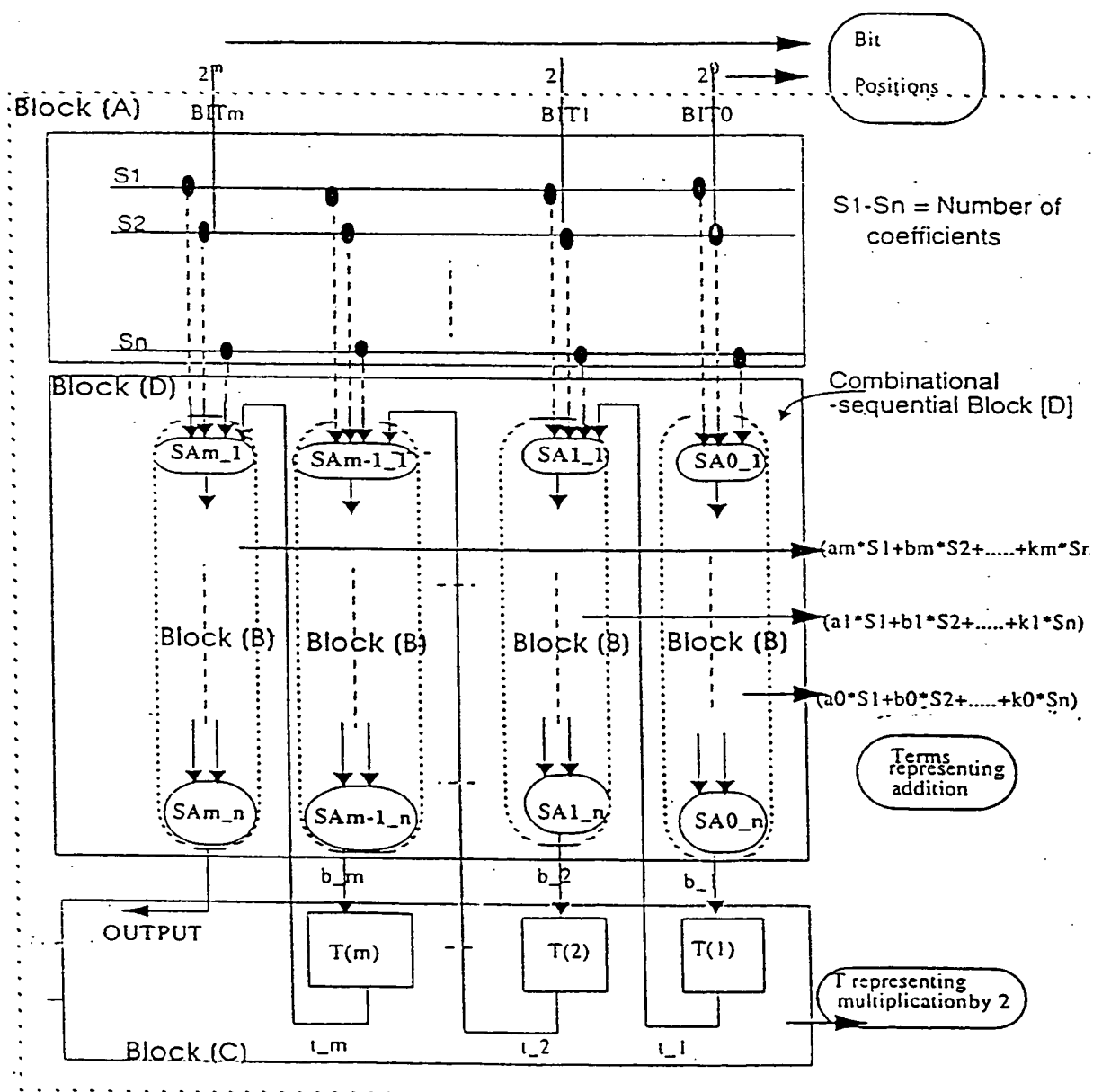   Approx. Number of flip-flops – number of coeff * (max. coefficient size/2)

FIGURE 9. Minimization (Already applied as patent)

FIGURE 10. Generalized structure "Minimization already applied as patent"

Approx. Number of serial adders = (number of coefficient
max coeff size /2)
Number of flip-flops(l) = Size of maximum coefficient

SUBSTITUTE SHEET (RULE 26)

FIGURE 11. Minimizations in "proposal for PATENT"

FIGURE 12. Generalized structure "proposal for PATENT"

FIGURE 13. MSBs of the Parallel output are directly available